



(11) **EP 2 254 311 B1**

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention
of the grant of the patent:
31.08.2011 Bulletin 2011/35

(51) Int Cl.:
H04L 29/12^(2006.01)

(21) Application number: **10176040.3**

(22) Date of filing: **15.06.2000**

(54) **Maintaining address translations for data communications**

Aufrechterhaltung von Adressenübertragungen für Datenkommunikationen

Maintien des translations d'adresses pour communications de données

(84) Designated Contracting States:
**AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE**

(30) Priority: **15.06.1999 US 333829**

(43) Date of publication of application:
24.11.2010 Bulletin 2010/47

(62) Document number(s) of the earlier application(s) in
accordance with Art. 76 EPC:
00936931.5 / 1 186 146

(73) Proprietor: **Tectia Oyj**
00520 Helsinki (FI)

(72) Inventors:
• **Kivinen, Tero**
02610 Espoo (FI)

• **Ylonen, Tatu**
02610 Espoo (FI)

(74) Representative: **Ruuskanen, Juha-Pekka**
Page White & Farrer
Bedford House
John Street
London
WC1N 2BF (GB)

(56) References cited:
• **NEWMAN DATA COMMUNICATIONS D:**
"Benchmarking Terminology for Firewall
Performance; draft-ietf-bmwg-secperf-07.txt"
IETF STANDARD-WORKING-DRAFT, INTERNET
ENGINEERING TASK FORCE, IETF, CH, vol.
bmwg, no. 7, 1 May 1999 (1999-05-01),
XP015016199 ISSN: 0000-0004

Note: Within nine months of the publication of the mention of the grant of the European patent in the European Patent Bulletin, any person may give notice to the European Patent Office of opposition to that patent, in accordance with the Implementing Regulations. Notice of opposition shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

EP 2 254 311 B1

Description

[0001] The invention relates in general to the field of secure communications between computers in a packet-switched data transmission networks. Morse particularly the invention relates to the field of setting up and maintaining secure communication connections through a Network Address Translation or protocol conversion.

[0002] The Internet Engineering Task Force (IETF) has standardized the IPSEC (Internet Protocol Security) protocol suite; the standards are well known from the Request For Comments or RFC documents number RFC2401, RFC2402, RFC2406, RFC2407, RFC2408 and RFC2409 mentioned in the appended list of references. The IPSEC protocols provide security for the IP or Internet Protocol, which itself has been specified in the RFC document number RFC791. IPSEC performs authentication and encryption on packet level by generating a new IP header, adding an Authentication Header (AH) or Encapsulating Security Payload (ESP) header in front of the packet. The original packet is cryptographically authenticated and optionally encrypted. The method used to authenticate and possibly encrypt a packet is identified by a security parameter index (SPI) value stored in the AH and ESP headers. The RFC document number RFC2401 specifies a transport mode and a tunnelling mode for packets; the present invention is applicable regardless of which of these modes is used.

[0003] In recent years, more and more vendors and Internet service providers have started performing network address translation (NAT). References to NAT are found at least in the RFC document number RFC 1631 as well as the documents which are identified in the appended list of references as Srisuresh98Terminology, SrisureshEgevang98, Srisuresh98Security, Holdrege-Srisuresh99, TYS99, Rekhter99, LoBorella99 and BorellaLo99. There are two main forms of address translation, illustrated schematically in Figs. 1a and 1b: host NAT 101 and port NAT 151. Host NAT 101 only translates the IP addresses in an incoming packet 102 so that an outgoing packet 103 has a different IP address. Port NAT 151 also touches the TCP and UDP port numbers (Traffic Control Protocol; User Datagram Protocol) in an incoming packet 152, multiplexing several IP addresses to a single IP address in an outgoing packet 153 and correspondingly demultiplexing a single IP address into several IP addresses for packets travelling in the opposite direction (not shown). Port NATs are especially common in the home and small office environment. The physical separation of input and output connections for the NAT devices is only shown in Figs. 1a and 1b for graphical clarity; in practice there are many possible ways for physically connecting a NAT.

[0004] Address translation is most frequently performed at the edge of a local network (i. e., translation between multiple local private addresses on one hand and fewer globally routable public addresses on the other). Most often, port NAT is used and there is only one

globally routable address. A local network 154 has been schematically illustrated in Fig. 1b. Such arrangements are becoming extremely commonplace in the home and small office markets. Some Internet service providers have also started giving private addresses to their customers, and perform address translation in their core networks for such addresses. In general, network address translation has been widely discussed in depth e. g. in the NAT working group within the Internet Engineering Task Force. The operating principles of a NAT device are well known, and there are many implementations available on the market from multiple vendors, including several implementations in freely available source code. The typical operation of a NAT may be described so that it maps IP address and port combinations to different IP address and port combinations. The mapping will remain constant for the duration of a network connection, but may change (slowly) with time. In practice, the NAT functionality is often integrated into a firewall or a router.

[0005] Fig. 1c illustrates an exemplary practical network communication situation where a transmitting node 181 is located in a first local area network (also known as the first private network) 182, which has a port NAT 183 to connect it to a wide-area general packet-switched network 184 like the Internet. The latter consists of a very large number of nodes interconnected in an arbitrary way. A receiving node 185 is located in a second local area network 186 which is again coupled to the wide-area network through a NAT 187. The denominations "transmitting node" and "receiving node" are somewhat misleading, since the communication required to set up network security services is bidirectional. The transmitting node is the one that initiates the communication. Also the terms "Initiator" and "Responder" are used for the transmitting node and the receiving node respectively.

[0006] The purpose of Fig. 1c is to emphasize the fact that the communicating nodes are aware of neither the number or nature of the intermediate devices through which they communicate nor the nature of transformations that take place. In addition to NATs, there are other types of devices on the Internet that may legally modify packets as they are transmitted. A typical example is a protocol converter, whose main job is to convert the packet to a different protocol without disturbing normal operation. Using them leads to problems very similar to the NAT case. A fairly simple but important example is converting between IPv4 and IPv6, which are different versions of the Internet Protocol. Such converters will be extremely important and commonplace in the near future. A packet may undergo several conversions of this type during its travel, and it is possible that the endpoints of the communication actually use a different protocol. Like NAT, protocol conversion is often performed in routers and firewalls.

[0007] It is well known in the IPSEC community that the IPSEC protocol does not work well across network address translations. The problem has been discussed at least in the references given as HoldregeSrisuresh99

and Rekhter99.

[0008] In the Finnish patent application number 974665 and the corresponding PCT application number F198/01032 we have presented a certain method for performing IPSEC address translations and a method for packet authentication that is insensitive to address transformations and protocol conversions en route of the packet. Additionally in said applications we have presented a transmitting network device and a receiving network device that are able to take advantage of the aforementioned method. However, some problems related to the provision of network security services over network address translation remain unsolved in said previous patent applications.

[0009] Network Working Group Internet-draft 'Benchmarking Terminology for Firewall Performance' of May 1999 by D. Newman, XP0150161991SSN, section 3.10 'Connection maintenance' describes use of "keep-alive" data to maintain a connection through a firewall in some implementations of TCP and other connection-oriented protocols during period where no user data is exchanged.

[0010] It is an object of the present invention to present a method and an apparatus for providing network network security services over network address translation in a reliable and advantageous way.

[0011] Method in accordance with the invention is defined in independent claims 1 and 2 and devices in accordance with the invention are defined in independent claims 8 and 9. More detailed aspects of the invention are defined in the dependent claims.

Fig. 1a illustrates the known use of a host NAT,

Fig. 1b illustrates the known use of a port NAT,

Fig. 1c illustrates a known communication connection between nodes through a packet-switched network,

Fig. 2a illustrates a certain Vendor ID payload applicable within the context of the invention,

Fig. 2b illustrates a certain private payload applicable within the context of the invention,

Fig. 2c illustrates a certain combined header structure applicable within the context of the invention,

Fig. 3 illustrates certain method steps related to the application of the invention,

Fig. 4 illustrates a transformation of header structures according to an aspect of the invention, and

Fig. 5 illustrates a simplified block diagram of a network device used to implement the method according to the invention.

[0012] The present invention combines and extends some of the methods of network address translation, tunneling over UDP, IKE, and the IKE extension mechanisms, in a novel and inventive way to produce a method for secure communications across network address translations and protocol conversions. The method can be made fully automatic and transparent to the user.

[0013] A key point relating to the applicability of the

invention is that-at the priority date of the present patent application-in general only TCP (described in RFC793) and UDP (described in RFC768) work over NAT. This is because most NATs used in practise are port NATs, and this is the form of NAT that provides most benefits with regards to the shortage of globally routable IP addresses. The invention is not, however, limited to the use of UDP and TCP as they are known at the priority date of this patent application: in general it may be said that UDP and TCP are examples of protocols that determine that connection identification information (i.e. addressing and port numbering) that is mapped into another form in the address transformation process. We may expect that other kinds of communication protocols and address transformations emerge in the future.

[0014] The various aspects of the invention are related to

- determining whether a remote host supports a certain method which is typically a secure communication method according to the invention (the "methods supported" aspect),
- determining what network address translations and/or protocol conversions occur on packets, if any (the "occurring translations" aspect),
- tunneling packets inside a certain carefully selected protocol, typically UDP, to make them traverse NATs (the "selected tunnelling" aspect),
- using a keepalive method to make sure that involved NAT devices and other devices that use timeouts for mappings do not lose the mapping for the communicating hosts (the "keepalive" aspect),
- compensating for the translations that occur before verifying the message authentication code for AH packets (the "compensation/authentication" aspect) and
- performing address translations at either the sending or receiving node to compensate for multiple hosts being mapped to a single public address (the "compensation/mapping" aspect).

[0015] The process of encapsulating data packets for transmission over a different logical network is called tunneling. Typically, in the case of the IP protocol, tunneling involves adding a new IP header in front of the original packet, setting the protocol field in the new header appropriately, and sending the packet to the desired destination (endpoint of the tunnel). Tunneling may also be implemented by modifying the original packet header fields or replacing them with a different header, as long as a sufficient amount of information about the original packet is saved in the process so that it will be possible to reconstruct the packet at the end of the tunnel into a form sufficiently similar to the original packet entering the tunnel. The exact amount of information that needs to be passed with the packet depends on the network protocols, and information may be passed either explicitly (as part of the tunnelled packet) or implicitly (by the context,

as determined e.g. by previously transmitted packets or a context identifier in the tunneled packet).

[0016] It is well known in the art how to tunnel packets over a network. At least the references given as RFC 1226, RFC 1234, RFC 1241, RFC 1326, RFC1701, RFC 1853, RFC2003, RFC2004, RFC2107, RFC2344, RFC2401, RFC2406, RFC2473 and RFC2529 relate to the subject of tunneling. For example, RFC 1234 presents a method of tunneling IPX frames over UDP. In that method, packets are tunneled to a fixed UDP port and to the decapsulator's IP address.

[0017] The IPSEC protocol mentioned in the background description typically uses the Internet Key Exchange or IKE protocol (known from references RFC2409, RFC2408 and RFC2407) for authenticating the communicating parties to each other, deriving a shared secret known only to the communicating parties, negotiating authentication and encryption methods to be used for the communication, and agreeing on a security parameter index (SPI) value and a set of selectors to be used for the communication. The IKE protocol was previously known as the ISAKMP/Oakley, where the acronym ISAKMP comes from Internet Security Association Key Management Protocol. Besides said normal negotiation specified in the IKE standard, IKE supports certain mechanisms for extension. The Vendor ID payload known from reference RFC2408 allows communicating parties to determine whether the other party supports a particular private extension mechanism. The IPSEC DOI (Domain of Interpretation) known as RFC2407 reserves certain numeric values for such private extensions.

[0018] Currently, the well-known Vendor ID payload is defined to have the format illustrated in Fig. 2a, where the column numbers correspond to bit positions. For the purposes of the present invention the Vendor ID field 201 is the most important part of the Vendor ID payload. In the context of the IKE protocol, negotiating whether the remote host supports a certain method for providing secure network communications can be performed as follows. The terminology used here is borrowed from the INE documents.

[0019] The IKE protocol determines the so-called Phase 1 of the mutual exchange of messages between the Initiator (i.e., the node first sending a packet to the other) and the Responder (i.e., the node first receiving a packet). Fig. 3 illustrates an exchange of first Phase 1 messages between the Initiator and the Responder. According to the "methods supported" aspect of the invention both devices include a certain Vendor ID Payload in a certain Phase 1 message which is most advantageously their first Phase 1 message. This payload indicates that they support the method in question. In Fig. 3 the Vendor ID fields contained within the Initiator's first (or other) Phase 1 message is schematically shown as 201' and the Vendor ID fields contained within the Responder's first (or other) Phase 1 message is schematically shown as 201". To indicate support for a certain method the Vendor ID field in the Vendor ID Payload is basically

an identification of that method: advantageously it is the MD5 hash of a previously known identification string, e.g. "SSH IPSEC NAT Traversal Version 1", without any trailing zeroes or newlines. Producing MD5 hashes of arbitrary character sequences is a technique well known in the art for example from the publication RFC1321 mentioned in the list of references.

[0020] Next we will address the "occurring translations" aspect of the invention. In addition to the above-mentioned Phase 1, the IKE protocol determines the so-called Phase 2 of the mutual exchange of messages between the Initiator and the Responder. According to the "occurring translations" aspect of the invention the parties can determine which translations occur by including the IP addresses they see in private payloads of certain Phase 2 Quick Mode messages, which are most advantageously their first Phase 2 Quick Mode messages. Any unused number in the private payload number range can be used to signify such use of the private payload (e.g. 157, which is unused at the priority date of the present patent application).

[0021] The private payload used to reveal the occurring translations can have e.g. the format illustrated in Fig. 2b. Field 211 contains a type code that identifies the types of the addresses that appear in fields 212 and 213. Field 212 contains the address of the Initiator as seen by the node sending the message, and field 213 contains the address of the Responder as seen by the node sending the message. Fig. 3 shows the exchange of (first) Phase 2 Quick Mode messages between the Initiator and the Responder so that the corresponding fields 211', 212' and 213' are included in the message sent by the former and the fields 211", 212" and 213" are included in the message sent by the latter.

[0022] According to known practice the addresses of the Initiator and Responder are also included in the header of the packet that contains the payload of Fig. 2b. In the header they are susceptible to address translations and other processing whereas in the private payload they are not. When the packet with the payload of Fig. 2b is received, the addresses contained in it are compared with those seen in the packet header. If they differ, then an address translation occurred on the packet. Later we will refer to the use of the standard IKE port number 500 together with applying the invention; as an additional way of detecting occurred translations the port numbers of the received packet can also be compared against the standard IKE port number 500 to determine if port translations occurred.

[0023] An aspect of some importance when handling the addresses is that the UDP source port of the packet can be saved for later use. It would usually be saved with the data structures for Phase 1 ISAKMP security associations, and would be used to set up compensation processing for Phase 2 IPSEC security associations.

[0024] To use the method described above to implement the "occurred translations" aspect of the invention, the hosts must modify their Phase 2 identification pay-

loads: the payload illustrated in Fig. 2b is not known in the existing standards. One possibility is to restrict the payloads to the ID_IPV4_ADDR and ID_IPV6_ADDR types, which would be appropriate for host-to-host operation.

[0025] Next we will address the "selected tunnelling", "compensation/authentication" and "compensation/mapping" aspects of the invention. According to this aspect of the invention the actual data packets can be tunneled over the same connection which is used to set up the security features of the communication connection, e.g. the UDP connection used for IKE. This ensures that the actual data packets will experience the same translations as the IKE packets did when the translation was determined. Taken that the standard port number 500 has been determined for IKE, this would mean that all packets are sent with source port 500 and destination port 500, and a method is needed to distinguish the real IKE packets from those containing encapsulated data. One possible way of doing this takes advantage of the fact that the IKE header used for real IKE packets contains an Initiator Cookie field: we may specify that Initiators that support this aspect of the invention never generate cookies that have all zeroes in their four first bytes. The value zero in the corresponding four bytes is then used to recognize the packet as a tunneled data packet. In this way, tunneled data packets would have four zero bytes at the beginning of the UDP payload, whereas real IKE packets never would.

[0026] Fig. 4 illustrates the encapsulation of actual IPSEC packets into UDP for transmission. Basically, a UDP header 403 and a short intermediate header 404 are inserted after the IP header 401 already in the packet (with the protocol field copied to the intermediate header). The IP header 401 is slightly modified to produce a modified IP header 401'. The IP payload 402 stays the same. The simple illustration of the unencapsulated IPSEC packet on the left should not be misinterpreted: this packet is not plaintext but has been processed according to AH or ESP or corresponding other transformation protocol in the sending node before its encapsulation into UDP.

[0027] Without limiting the generality, it is assumed in the presentation here that the encapsulation according to Fig. 4 is always performed by the same nodes that perform IPSEC processing (either an end node or a VPN device). It should also be noted that instead of encapsulating the IPSEC packets into UDP they could be encapsulated into TCP. This alternative would probably require using fake session starts and ends so that the first packet has the SYN bit and the last packet has the FIN bit, as specified in the TCP protocol.

[0028] In encapsulating an actual data packet or a "datagram" according to Fig. 4, the original IP header 401 - defined in RFC791 - is modified to produce the modified IP header 401' as follows:

* the Protocol field in the IP header (not separately shown) is replaced by protocol 17 for UDP in accordance with RFC768,

* the Total Length field in the IP header (not separately shown) is incremented by the combined size of the UDP and intermediate headers (total 16 bytes) and

* the Header Checksum field in the IP header (not separately shown) is recomputed in accordance with the rules given in RFC791.

[0029] As seen from Fig. 4, an UDP header 403 - as defined in RFC768 - and an intermediate header 404 are inserted after the IP header. The UDP header is 8 octets and the intermediate header is 8 octets, for a total of 16 octets. These headers are treated as one in the following discussion. The combined header has most advantageously the format illustrated in Fig. 2c. Fields of this header are set as follows:

* The Source Port field 221 is set to 500 (same as IKE). If the packet goes through NAT, this may be different when the packet is received.

* The Destination Port field 222 is set to the port number from which the other end appears to be sending packets. If the packet goes through NAT, the recipient may see a different port number here.

* The UDP Length field 223 is the length of the UDP header plus the length of the UDP data field. In this case, it also includes the intermediate header. The value is computed in bytes as 16 plus the length of the original IP packet payload (not including the original IP header, which is included in the Length field in the IP header).

* The UDP Checksum field 224 is most advantageously set to 0. The UDP checksum is optional, and we do not wish to calculate or check it with this tunneling mechanism. Integrity of the data is assumed to be protected by an AH or ESP header within the tunneled packet.

* The Must be zero field 225: This field must contain a previously agreed fixed value, which is most advantageously all zeroes. The field overlaps with the first four bytes of the Initiator Cookie field in an actual IKE header. Any Initiator that supports this aspect of the invention must not use a cookie where the first four bytes are zero. These zero bytes are used to separate the tunneled packets from real ISAKMP packets. Naturally some other fixed value than "all zeroes" could be chosen, but the value must be fixed for this particular use.

* Protocol field 226: The value of this field is copied from the known Protocol field in the original IP header

(not separately shown in Fig. 4).

* Reserved field 227: most advantageously sent as all zeroes; ignored on reception.

[0030] The sender inserts this header in any packets tunneled to a destination behind NAT. Information about whether NAT is used can be stored on a per SA (Security Association) basis in the policy manager. The encapsulation referred to in Fig. 4 can be implemented either as a new transform or as part of the otherwise known AH and ESP transforms.

[0031] The encapsulation operation makes use of the UDP port number and IP address of the remote host, which were determined during the IKE negotiation.

[0032] The receiver decapsulates packets from this encapsulation before doing AH or ESP processing. Decapsulation removes this header and updates the Protocol, Length, and Checksum fields of the IP header. No configuration data (port number etc.) is needed for this operation.

[0033] The decapsulation should be performed only if all of the following selectors match:

* destination address is the destination address of this host,

* source address is the address of a host with which this host has agreed to use this tunnelling,

* the Protocol field indicates UDP,

* the Destination port field value is 500 and

* the Source port field value indicates the port with which this host has agreed to use this tunneling. (Note that there may be multiple source addresses and ports for which this tunneling is performed; each of them is treated by a separate set of selectors.)

[0034] During decapsulation the source address in the received packet can be replaced by the real source address received during the IKE negotiation. This implements the compensation for AH MAC verification. The address is again changed in the post-processing phase below. Because of this compensation, the standard AH and ESP transforms can be used unmodified.

[0035] In Fig. 3 the AH/ESP processing at the sending node is schematically shown as block 301, encapsulation of datagrams into UDP is schematically shown as block 302, the corresponding decapsulation of datagrams from UDP is schematically shown as block 303 and AH/ESP processing at the receiving node is schematically shown as block 304.

[0036] Additional compensation must be done after the packet has been decapsulated from AH or ESP. This additional decapsulation must deal with the fact that the outer packet actually went through NAT (illustrated sche-

matically in Fig. 3 as block 305), and consequently the plaintext packet must also undergo a similar transformation. The recipient must see the address of the NAT device as the address of the host, rather than the original internal address. Alternatively, this compensation could have been performed by the sender of the packet before encapsulating it within AH or ESP.

[0037] There are several alternatives for this additional compensation for various special cases (the best compensation depends on the particular application):

* Allocating a range of network addresses for this processing (say, in the link-local use range 169.254.x.x - the actual values do not matter; basically we just want an arbitrary network that no-one else is using). An address in this range is allocated for each <natip, ownip, natport, ownport> combination, where natip means the IP address of the NAT, ownip means the processing device's own IP address, natport means the port number at the NAT and ownport means the processing device's own port number. The remote address in the packet is replaced by this address before the packet is sent to protocol stacks.

* As part of the compensation, the TCP checksum for internal hosts must be recomputed if host addresses or port numbers changed. TCP checksum computations may also be incremental, as is known from RFC1071. Port NAT may need to be performed for the source port.

* When used as a VPN between two sites using incompatible (possibly overlapping) private address spaces, address translation must be performed to make the addresses compatible with local addresses.

* When used as a VPN between two sites using compatible (non-overlapping) private address spaces, and tunnel mode is used, no additional compensation may be needed.

* Address translation may need to be performed for the contents of certain protocol packets, such as FTP (known from RFC959) or H.323. Other similar issues are discussed in the reference given as HoldregeSrisuresh99.

* It may also be possible to use random addresses for the client at the server, and perform address translation to this address. This could allow the server to distinguish between multiple clients behind the same NAT, and could avoid manual configuration of the local address space.

* The compensation operation may or may not interact with the TCP/IP stack on the local machine to

reserve UDP port numbers.

[0038] In general, this invention does not significantly constrain the method used to compensate for inner packets the NAT occurring for the outer header. The optimal method for performing such compensation may be found among the above-given alternatives by experimenting, or some other optimal method could be presented.

[0039] Next we will address the "keepalive" aspect of the invention, i.e. ensuring that the network address translations performed in the network do not change after the translations that occur have been determined. Network address translators cache the information about address mapping, so that they can reverse the mapping for reply packets. If TCP is used, the address translator may look at the FIN bit of the TCP header to determine when it can drop a particular mapping. For UDP, however, there is no explicit termination indication for flows. For this reason, many NATs will time out mappings for UDP quite fast (even as fast as in 30 seconds). Thus, it becomes necessary to force the mapping to be maintained.

[0040] A possible way of ensuring the maintaining of mappings is to send keepalive packets frequently enough that the address translation remains in the cache. When computing the required frequency, one must take into account that packets may be lost in the network, and thus multiple keepalives must be sent within the estimated shortest period in which NATs may forget the mapping. The appropriate frequency depends on both the period the mappings are kept cached and on the packet loss probability of the network; optimal frequency values for various context may be found through experimenting.

[0041] Keepalive packets do not need to contain any meaningful information other than the necessary headers that are equal to the data packet headers to ensure that the keepalive packets will be handled exactly in the same way as the actual data packets. A keepalive packet may contain an indicator that identifies it as a keepalive packet and not a data packet; however it may also be determined that all packets that do not contain meaningful payload information are interpreted to be keepalive packets. In Fig. 3 the transmission of keepalive packets is schematically illustrated by block 306 and the reception and discarding of them is schematically illustrated by block 307. It should be noted that the use of keepalive packets is not needed at all if actual data packets are transmitted frequently enough and/or the connection is to remain valid only for such a short time (e.g. a few seconds) that it is improbable that any intermediate device would delete the mapping information from its cache. Keepalive packets need to be transmitted in one direction only, although they may be transmitted also bidirectionally; the drawback resulting from their bidirectional transmission is the resulting increase in unnecessary network traffic. The invention does not limit the direction(s) in which keepalive packets (if any) are transmitted.

[0042] Fig. 5 is a simplified block diagram of a network device 500 that can act as the Initiator or the Responder

according to the method of providing secure communications over network address translations in accordance with the invention. Network interface 501 connects the network device 500 physically to the network. Address management block 502 keeps track of the correct network addresses, port numbers and other essential public identification information of both the network device 500 itself and its peer (not shown). IKE block 503 is responsible for the key management process and other activities related to the exchange of secret information. Encryption/decryption block 504 implements the encryption and decryption of data once the secret key has been obtained by the IKE block 503.

[0043] Compensation block 505 is used to compensate for the permissible transformations in the transmitted and/or received packets according to the invention. Either one of blocks 504 and 505 may be used to transmit, receive and discard keepalive packets. Packet assembler/disassembler block 506 is the intermediary between blocks 502 to 505 and the physical network interface 501. All blocks operate under the supervision of a control block 507 which also takes care of the routing of information between the other blocks and the rest of the network device, for example for displaying information to the user through a display unit (not shown) and obtaining commands from the user through a keyboard (not shown). The blocks of Fig. 5 are most advantageously implemented as pre-programmed operational procedures of a microprocessor, which implementation is known as such to the person skilled in the art. Other arrangements than that shown in Fig. 5 may as well be used to reduce the invention into practice.

[0044] Even though the present invention was presented in the context of IKE, and tunneling using the IKE port, it should be understood that the invention applies to also other analogous cases using different packet formatting methods, different negotiation details, a different key exchange protocol, or a different security protocol. The invention may also be applicable to non-IP protocols with suitable characteristics. The invention is equally applicable to both IPv4 and IPv6 protocols. The invention is also intended to apply to future revisions of the IPSEC and IKE protocols.

[0045] It should also be understood that the invention can also be applied to protocol translations in addition to just address translations. Adapting the present invention to protocol translations should be well within the capabilities of a person skilled in the art given the description here and the discussions regarding protocol translation in the former patent applications of the same applicant mentioned above.

LIST OF REFERENCES

[0046] BorellaLo99

M. Borella, J. Lo: Realm Specific IP: Protocol Specification, draft-ietf-nat-rsip-protocol-00.txt, Work in

Progress, Internet Engineering Task Force, 1999.
HoldregeSrisuresh99

M. Holdrege, P. Srisuresh: Protocol Complications
with the IP Network Address Translator (NAT), draft-
ietf-nat-protocol-complications-00.txt(Work in
Progress, Internet Engineering Task Force, 1999.
LoBorella99

J. Lo, M. Borella: Real Specific IP: A Framework,
draA-ietf-nat-rsip-framework-00.txt, Work in
Progress, Internet Engineering Task Force, 1999.
Rekhter99

Y. Rekhter: Implications of NATs on the TCP/IP ar-
chitecture, draft-ietf-nat-arch-implications-00.txt, In-
ternet Engineering Task Force, 1999.
RFC768

J. Postel: User Datagram Protocol, RFC 768, Inter-
net Engineering Task Force, 1980.
RFC791

J. Postel: Internet Protocol, RFC 791, Internet Engi-
neering Task Force, 1981.
RFC793

J. Postel: Transmission Control Protocol, RFC 793,
Internet Engineering Task Force, 1981.
RFC959

J. Postel, J. Reynolds: File Transfer Protocol, RFC
959, Internet Engineering Task Force, 1985.
RFC1071

R. Braden, D. Borman, C. Partridge: Computing the
Internet checksum, RFC 1071, Internet Engineering
Task Force, 1988.
RFC 1226

B. Kantor: Internet protocol encapsulation of AX.25
frames, RFC 1226, Internet Engineering Task Force,
1991.
RFC 1234

D. Provan: Tunneling IPX traffic through IP networks,
RFC 1234, Internet Engineering Task Force, 1991.
RFC 1241

R. Woodburn, D. Mills: Scheme for an internet en-
capsulation protocol: Version 1, RFC 1241, Internet
Engineering Task Force, 1991.
RFC1321

R. Rivet The MD5 message-digest algorithm, RFC
1321, Internet Engineering Task Force, 1992.
RFC1326

P. Tsuchiya: Mutual Encapsulation Considered Dan-
gerous, RFC 1326, Internet Engineering Task Force,
1992.
RFC 1631

K. Egevang, P. Francis: The IP Network Address
Translator (NAT), RFC 1631, Internet Engineering
Task Force, 1994.
RFC 1701

S. Hanks, T. Li, D. Farinacci, P. Traina: Generic
Routing Encapsulation, RFC 1701, Internet Engi-
neering Task Force, 1994.
RFC 1702

S. Hanks, T. Li, D. Farinacci, P. Traina: Generic
Routing Encapsulation over IPv4 networks, RFC
1702, Internet Engineering Task Force, 1994.
RFC 1853

W. Simpson: IP in IP Tunneling, RFC 1853, Internet
Engineering Task Force, 1995.
RFC2003

C. Perkins: IP Encapsulation within IP, RFC 2003,
Internet Engineering Task Force, 1996.
RFC2004

C. Perkins: IP Encapsulation within IP, RFC 2004,
Internet Engineering Task Force, 1996.
RFC2107

K. Hamzeh: Ascend Tunnel Management Protocol,
RFC 2107, Internet Engineering Task Force, 1997.
RFC2344

G. Montenegro: Reverse Tunneling for Mobile IP,
FC 2344, Internet Engineering Task Force, 1998.
RFC2391

P. Srisuresh, D. Gan: Load Shearing using IP Net-
work Address Translation (LSNAT), RFC 2391, In-
ternet Engineering Task Force, 1998.
RFC2401 1

S. Kent, R Atkinson: Security Architecture for the In-
ternet Protocol, RFC 2401, Internet Engineering
Task Force, 1998.
RFC2402

S. Kent, R Atkinson: IP Authentication Header, RFC
2402, Internet Engineering Task Force, 1998.
RFC2406

S. Kent, R. Atkinson: IP Encapsulating Security Pay-
load, RFC 2406, Internet Engineering Task Force,
1998.
RFC2407

D. Piper: The Internet IP Security Domain of Interpretation for ISAKMP. RFC 2407, Internet Engineering Task Force, 1998.
RFC2408

D. Maughan, M. Schertler, M. Schneider, J. Turner: Internet Security Association and Key Management Protocol (ISAKMP), RFC 2408, Internet Engineering Task Force, 1998.
RFC2409

D. Hakins, D. Carrel: The Internet Key Exchange (IKE), RFC 2409, Internet Engineering Task Force, 1998.
RFC2473

A. Conta, S. Deering: Generic Packet Tunneling in IPv6 Specification, RFC 2473, Internet Engineering Task Force, 1998.
RFC2529

B. Carpenter, C. Jung: Transmission of IPv6 over IPv4 Domains without Explicit Tunnels, RFC 2529, Internet Engineering Task Force, 1999.
Srisuresh98Terminology

P. Srisuresh: IP Network Address Translator (NAT) Terminology and Considerations, draft-ietf-nat-terminology-01.txt, Work in Progress, Internet Engineering Task Force, 1998.
Srisuresh98Security

P. Srisuresh: Security Model for Network Address Translator (NAT) Domains, draft-ietf-nat-seurity-01.txt, Work in Progress, Internet Engineering Task Force, 1998.
SrisureshEgevang98

P. Srisuresh, K. Egevang: Traditional IF Network Address Translator (Traditional NAT), draft-ietf-nat-traditional-01.txt, Work in Progress, Internet Engineering Task Force, 1998.
TYS99

W. Teo, S. Yeow, R Singh: IP Relocation through twice Network Address Translators (RAT), draft-ietf-nat-mat-00.txt, Work in Progress, Internet Engineering Task Force, 1999.

Claims

1. A method of maintaining communication of datagrams in a communication system where address translation is provided by a network address translator (305) for communication of datagrams between a first device and a second device, **characterised by** maintaining a determined network address trans-

lation for communication of datagrams between the first device and the second device by sending (306) from the first device or the second device at least one keepalive packet before a time out of the determined network address translation.

2. A method of providing address translations by a network address translator (305), comprising determining an address translation for communication of datagrams in between a first device and a second device; **characterised by** receiving at least one keepalive packet from the first device and/or the second device before time out of the determined address translation for the communication of datagrams; and in response to receiving the at least one keepalive packet, maintaining the determined network address translation for the communication of datagrams between the first device and the second device.

3. A method according to claim 1 or 2, wherein the at least one keepalive packet comprises a header that equals with the headers of the datagrams.

4. A method according to any preceding claim, wherein the at least one keepalive packet contains an indicator that identifies it as a keepalive packet.

5. A method according to any preceding claim, wherein a packet is interpreted as a keepalive packet if it does not contain any meaningful payload.

6. A method according to claim 1 or any claim dependent on claim 1, comprising determining a shortest period for the time out, and based on the determination, sending the at least one keepalive packet frequently enough to maintain the determined address translation in the network address translator (305).

7. A method according to claim 1 or any claim dependent on claim 1, comprising taking the possibility of packet loss into account in determining the frequency of sending the at least one keepalive packet.

8. A device (500) for communication of datagrams in a communication system where address translation is provided by a network address translator (305) for communication of datagrams between the device and a second device, **characterised by** means (504 or 505) for maintaining a determined network address translation for the communication of datagrams between the device and the second device by causing sending of at least one keepalive packet before a time out of the determined network address translation.

9. A device for network address translations (305), comprising

means for determining an address translation for communication of datagrams between a first device and a second device; **characterised by** means for maintaining the determined network address translation for the communication of datagrams between the first device and the second device in response to reception of at least one keepalive packet from the first device and/or the second device before time out of the determined address translation for the communication of datagrams.

10. A device according to claim 8 or 9, wherein the at least one keepalive packet comprises a header that equals with the headers of the datagrams.
11. A device according to any of claims 8 to 10, wherein the at least one keepalive packet contains an indicator that identifies it as a keepalive packet.
12. A device according to any of claims 8 to 11, wherein the device is configured to interpret a packet as a keepalive packet if the packet does not contain any meaningful payload.
13. A device (500) according to claim 8 or any claim dependent on claim 8, wherein the means for maintaining is configured to cause sending of the at least one keepalive packet frequently enough to maintain the determined address translation.
14. A device (500) according to claim 8 or any claim dependent on claim 8, wherein the means for maintaining is configured to take into account the possibility of packet loss in determining the frequency of sending.
15. A computer program comprising program code means adapted to perform any of steps of any of claims 1 to 7 when the program is run on a processor.

Patentansprüche

1. Verfahren zum Aufrechterhalten einer Kommunikation von Datagrammen in einem Kommunikationssystem, in welchem Adressen-Umsetzung durch einen Netzwerkadressenumsetzer (305) zur Kommunikation von Datagrammen zwischen einer ersten Vorrichtung und einer zweiten Vorrichtung bereitgestellt wird, **gekennzeichnet durch** das Aufrechterhalten einer vorbestimmten Netzwerkadressenumsetzung zur Kommunikation von Datagrammen zwischen der ersten Vorrichtung und der zweiten Vorrichtung **durch** Versenden (306) zumindest eines Aufrechterhaltungspakets von der ersten Vorrichtung oder der zweiten Vorrichtung vor einer Unterbrechung der vorbestimmten Netzwerkadressenumsetzung.

2. Verfahren zum Bereitstellen von Adressenumsetzungen durch einen Netzwerkadressenumsetzer (305), mit

- Bestimmen einer Adressenumsetzung für die Kommunikation von Datagrammen zwischen einer ersten Vorrichtung und einer zweiten Vorrichtung, **gekennzeichnet durch**
- Empfangen zumindest eines Aufrechterhaltungspakets von der ersten Vorrichtung und/oder der zweiten Vorrichtung vor einer Unterbrechung der vorbestimmten Adressenumsetzung für die Kommunikation von Datagrammen; und
- in Antwort auf das Empfangen des zumindest einen Aufrechterhaltungspakets, Aufrechterhalten der vorbestimmten Netzwerkadressenumsetzung für die Kommunikation von Datagrammen zwischen der ersten Vorrichtung und der zweiten Vorrichtung.

3. Verfahren nach Anspruch 1 oder 2, wobei das zumindest eine Aufrechterhaltungspaket einen Header umfasst, der den Headern der Datagramme entspricht.
4. Verfahren nach einem der vorstehenden Ansprüche, wobei das zumindest eine Aufrechterhaltungspaket einen Indikator enthält, der dieses als Aufrechterhaltungspaket kennzeichnet.
5. Verfahren nach einem der vorstehenden Ansprüche, wobei ein Paket als Aufrechterhaltungspaket interpretiert wird, wenn es keine sinnvolle Nutzlast enthält.
6. Verfahren nach Anspruch 1 oder nach einem von Anspruch 1 abhängigen Anspruch, mit dem Bestimmen einer kürzesten Periode für die Unterbrechung, und basierend auf der Bestimmung, Senden des zumindest einen Aufrechterhaltungspakets in ausreichender Frequenz, um die vorbestimmte Adressenumsetzung in der Netzwerkadressenumsetzung (305) aufrecht zu erhalten.

7. Verfahren nach Anspruch 1 oder nach einem von Anspruch 1 abhängigen Anspruch, mit dem in Betracht ziehen der Möglichkeit eines Paketverlustes bei der Bestimmung der Frequenz des Sendens des zumindest einen Aufrechterhaltungspakets.
8. Vorrichtung (500) zur Kommunikation von Datagrammen in einem Kommunikationssystem, in welchem Adressenumsetzung durch einen Netzwerkadressenumsetzer (305) zur Kommunikation von Datagrammen zwischen der Vorrichtung und einer zweiten Vorrichtung bereitgestellt wird, **gekennzeichnet durch** Mittel (504 oder 505) zum Aufrecht-

erhalten einer vorbestimmten Netzwerkadressenumsetzung für die Kommunikation von Datagrammen zwischen der Vorrichtung und der zweiten Vorrichtung **durch** Verursachen der Sendung von zumindest einem Aufrechterhaltungspakets vor einer Unterbrechung der vorbestimmten Netzwerkadressenumsetzung.

9. Vorrichtung für Netzwerkadressenumsetzung (305), mit

- Mitteln zum Bestimmen einer Adressenumsetzung für eine Kommunikation von Datagrammen zwischen einer ersten Vorrichtung und einer zweiten Vorrichtung; **gekennzeichnet durch**

- Mittel zum Aufrechterhalten der vorbestimmten Netzwerkadressenumsetzung für die Kommunikation von Datagrammen zwischen der ersten Vorrichtung und der zweiten Vorrichtung in Antwort auf den Empfang von zumindest einem Aufrechterhaltungspaket von der ersten Vorrichtung und/oder der zweiten Vorrichtung vor einer Unterbrechung der vorbestimmten Adressenumsetzung für die Kommunikation von Datagrammen.

10. Vorrichtung nach Anspruch 8 oder 9, wobei das zumindest eine Aufrechterhaltungspaket einen Header umfasst, der den Headern der Datagramme entspricht.

11. Vorrichtung nach einem der Ansprüche 8 bis 10, wobei das zumindest eine Aufrechterhaltungspaket einen Indikator enthält, der dieses als Aufrechterhaltungspaket kennzeichnet.

12. Vorrichtung nach einem der Ansprüche 8 bis 11, wobei die Vorrichtung dazu ausgestaltet ist, ein Paket als Aufrechterhaltungspaket zu interpretieren, wenn das Paket keine sinnvolle Nutzlast enthält.

13. Vorrichtung (500) nach Anspruch 8 oder nach einem von Anspruch 8 abhängigen Anspruch, wobei die Mittel zum Aufrechterhalten dazu ausgestaltet sind, eine Sendung des zumindest einen Aufrechterhaltungspakets in ausreichender Frequenz zu verursachen, um die vorbestimmte Adressenumsetzung aufrecht zu erhalten.

14. Vorrichtung (500) nach Anspruch 8 oder nach einem von Anspruch 8 abhängigen Anspruch, wobei die Mittel zum Aufrechterhalten dazu ausgestaltet sind, die Möglichkeit eines Paketverlustes bei der Bestimmung der Sendefrequenz in Betracht zu ziehen.

15. Computerprogramm mit einem Programmcodemittel, welches dazu geeignet ist, einen der Schritte der

Ansprüche 1 bis 7 auszuführen, wenn das Programm auf einem Prozessor läuft.

5 Revendications

1. Procédé de maintien de communication de datagrammes dans un système de communication où une traduction d'adresse est fournie par un traducteur d'adresse de réseau (305) pour la communication de datagrammes entre un premier dispositif et un second dispositif, **caractérisé par** le maintien d'une traduction d'adresse de réseau déterminée pour la communication de datagrammes entre le premier dispositif et le second dispositif en envoyant (306) à partir du premier dispositif ou du second dispositif au moins un paquet d'entretien avant une temporisation de la traduction d'adresse de réseau déterminée.

2. Procédé de fourniture de traduction d'adresses par un traducteur d'adresse de réseau (305), comprenant l'étape consistant à :

déterminer une traduction d'adresse pour la communication de datagrammes entre un premier dispositif et un second dispositif ; **caractérisé par** les étapes consistant à :

recevoir au moins un paquet d'entretien à partir du premier dispositif et/ou du second dispositif avant la temporisation de la traduction d'adresse déterminée pour la communication de datagrammes ; et
en réponse à l'étape consistant à recevoir l'au moins un paquet d'entretien, à maintenir la traduction d'adresse de réseau déterminée pour la communication de datagrammes entre le premier dispositif et le second dispositif.

3. Procédé selon la revendication 1 ou 2, dans lequel l'au moins un paquet d'entretien comprend un en-tête qui est égal aux en-têtes des datagrammes.

4. Procédé selon une quelconque revendication précédente, dans lequel l'au moins un paquet d'entretien contient un indicateur qui l'identifie comme étant un paquet d'entretien.

5. Procédé selon une quelconque revendication précédente, dans lequel un paquet est interprété comme étant un paquet d'entretien s'il ne contient aucune donnée utile significative.

6. Procédé selon la revendication 1 ou une quelconque revendication dépendant de la revendication 1, comprenant les étapes consistant à déterminer une pé-

- riode la plus courte pour la temporisation, et, en fonction de la détermination, à envoyer l'au moins un paquet d'entretien suffisamment fréquemment pour maintenir la traduction d'adresse déterminée dans le traducteur d'adresse de réseau (305). 5
7. Procédé selon la revendication 1 ou une quelconque revendication dépendant de la revendication 1, comprenant l'étape consistant à prendre la possibilité de perte de paquet en compte dans la détermination de la fréquence de l'envoi de l'au moins un paquet d'entretien. 10
8. Dispositif (500) pour communication de datagrammes dans un système de communication où une traduction d'adresse est fournie par un traducteur d'adresse de réseau (305) pour la communication de datagrammes entre le dispositif et un second dispositif, **caractérisé par** un moyen (504 ou 505) pour maintenir une traduction d'adresse de réseau déterminée pour la communication de datagrammes entre le dispositif et le second dispositif en entraînant l'envoi d'au moins un paquet d'entretien avant une temporisation de la traduction d'adresse de réseau déterminée. 20 25
9. Dispositif de traduction d'adresses de réseau (305), comprenant :
- un moyen pour déterminer une traduction d'adresse pour la communication de datagrammes entre un premier dispositif et un second dispositif ; **caractérisé par** 30
- un moyen pour maintenir la traduction d'adresse de réseau déterminée pour la communication de datagrammes entre le premier dispositif et le second dispositif en réponse à la réception d'au moins un paquet d'entretien à partir du premier dispositif et/ou du second dispositif avant une temporisation de la traduction d'adresse déterminée pour la communication de datagrammes. 35 40
10. Dispositif selon la revendication 8 ou 9, dans lequel l'au moins un paquet d'entretien comprend un en-tête qui est égal aux en-têtes des datagrammes. 45
11. Dispositif selon une quelconque des revendications 8 à 10, dans lequel l'au moins un paquet d'entretien contient un indicateur qui l'identifie comme étant un paquet d'entretien. 50
12. Dispositif selon une quelconque des revendications 8 à 11, dans lequel le dispositif est configuré pour interpréter un paquet comme étant un paquet d'entretien si le paquet ne contient aucune donnée utile significative. 55
13. Dispositif (500) selon la revendication 8 ou une quelconque revendication dépendant de la revendication 8, dans lequel le moyen pour maintenir est configuré pour entraîner l'envoi de l'au moins un paquet d'entretien suffisamment fréquemment pour maintenir la traduction d'adresse déterminée.
14. Dispositif (500) selon la revendication 8 ou une quelconque revendication dépendant de la revendication 8, dans lequel le moyen pour maintenir est configuré pour prendre en compte la possibilité de perte de paquet dans la détermination de la fréquence d'envoi.
15. Programme informatique comprenant des moyens de code de programme adaptés pour réaliser une quelconque des étapes selon une quelconque des revendications 1 à 7 lorsque le programme est exécuté sur un processeur.

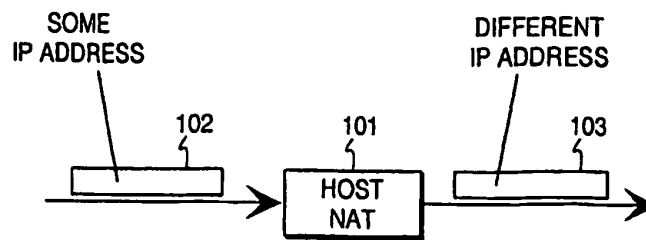


Fig. 1a
PRIOR ART

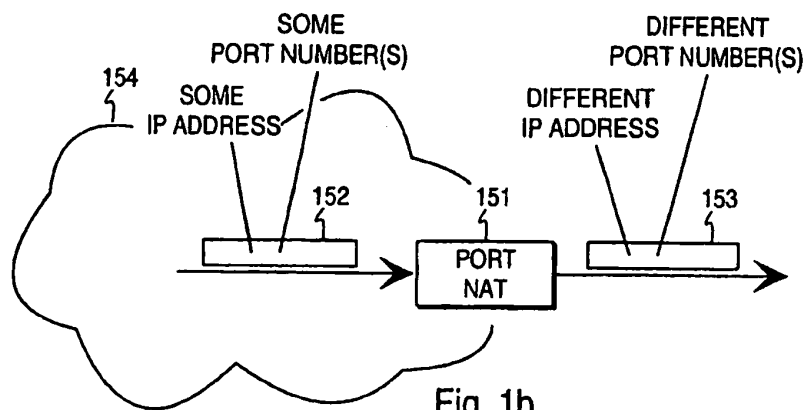


Fig. 1b
PRIOR ART

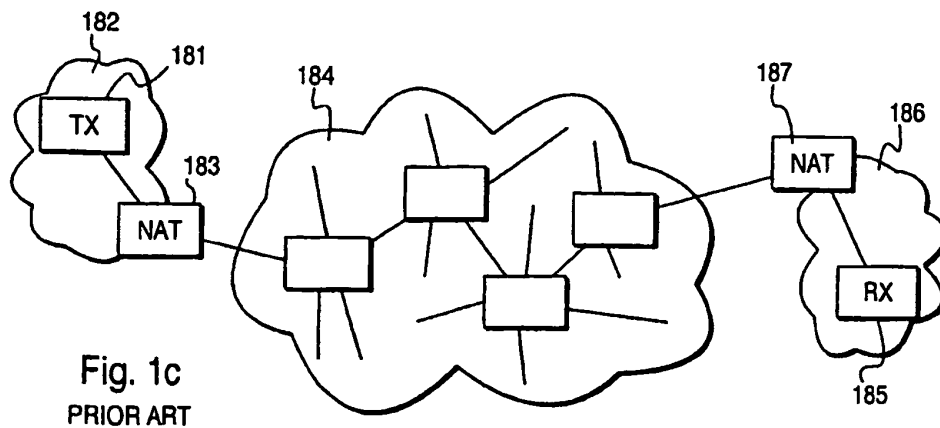


Fig. 1c
PRIOR ART

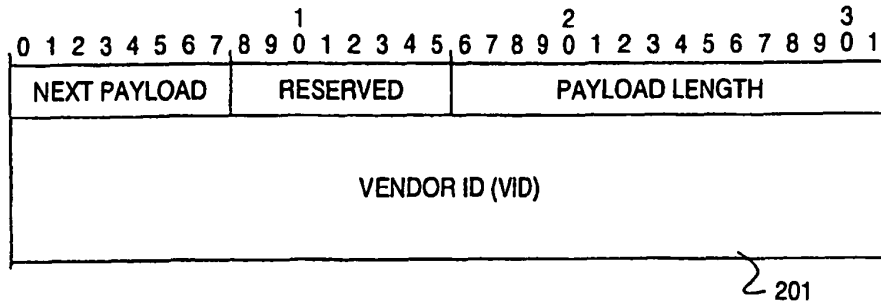


Fig. 2a

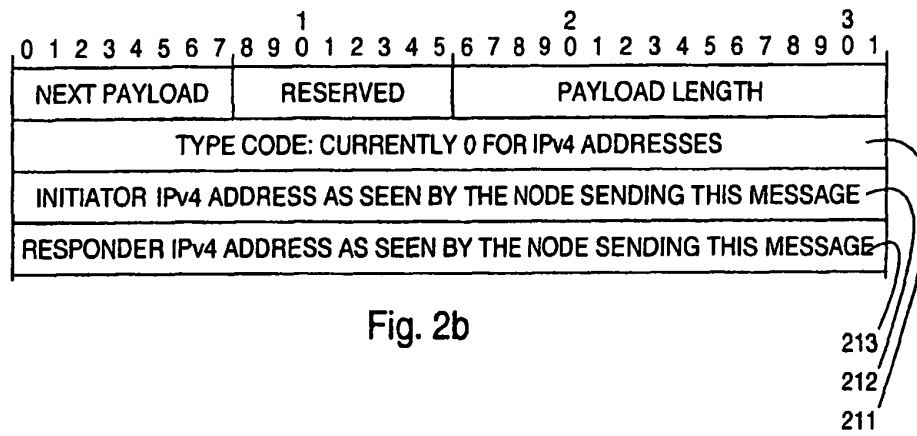


Fig. 2b

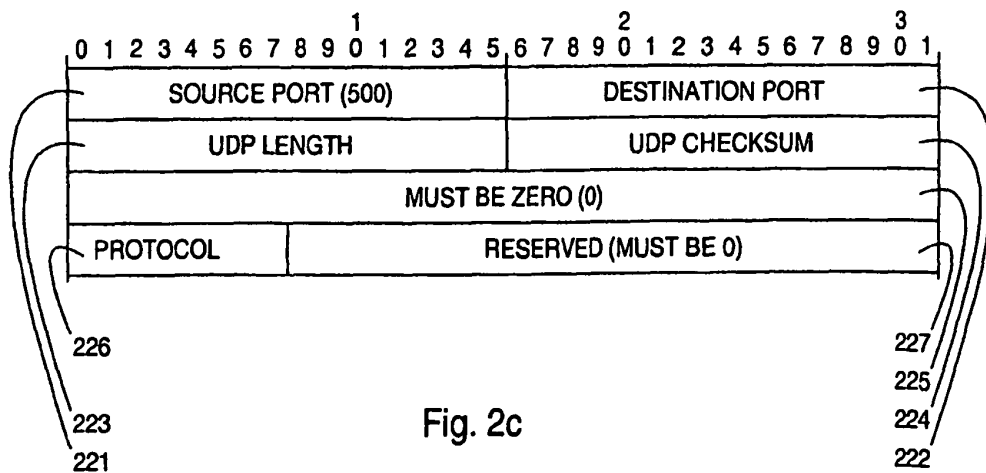


Fig. 2c

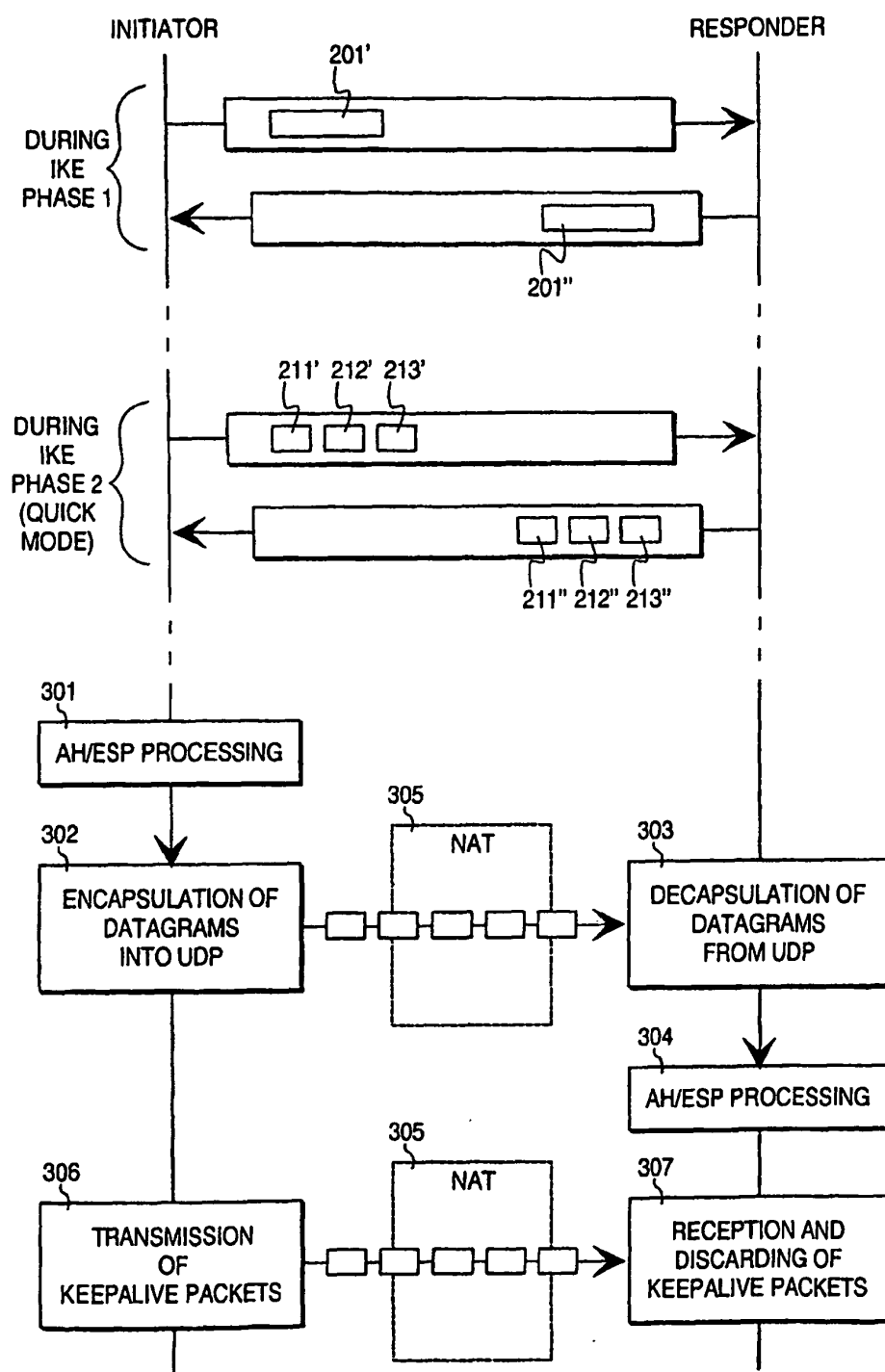


Fig. 3

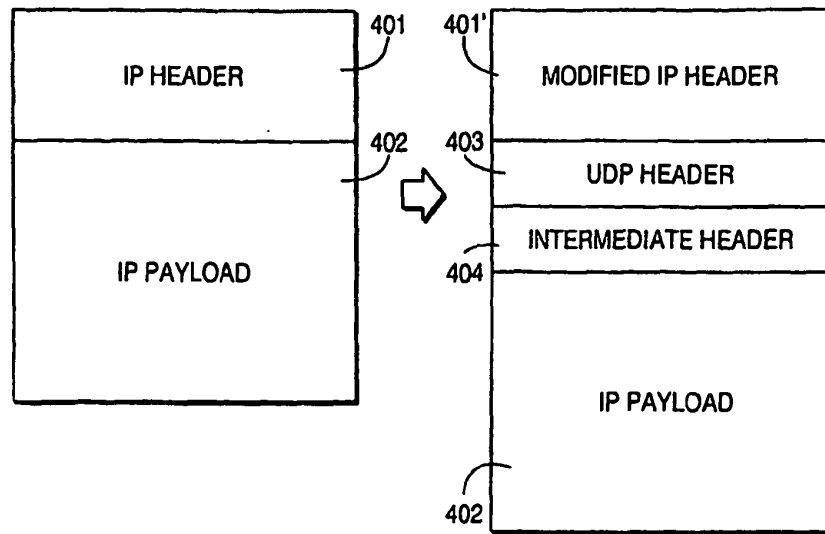


Fig. 4

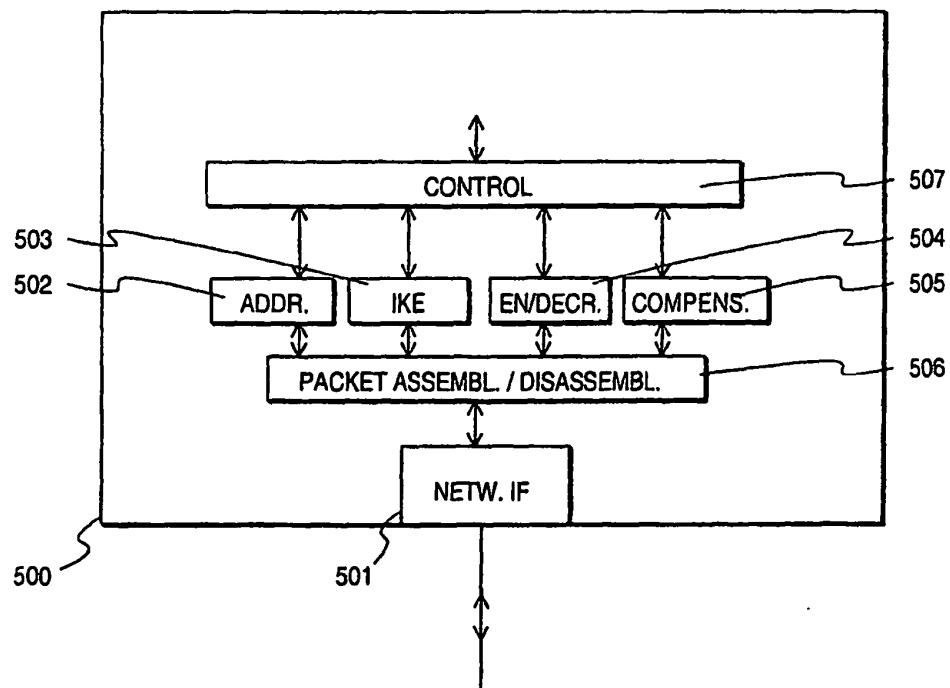


Fig. 5

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- FI 974665 [0008]
- FI F19801032 W [0008]

Non-patent literature cited in the description

- **D. Newman.** Benchmarking Terminology for Firewall Performance. *Network Working Group Internet-draft*, May 1999 [0009]
- **M. Borella ; J. Lo.** Realm Specific IP: Protocol Specification, draft-ietf-nat-rsip-protocol-00.txt. *Work in Progress, Internet Engineering Task Force*, 1999 [0046]
- **M. Holdrege ; P. Srisuresh.** Protocol Complications with the IP Network Address Translator (NAT), draft-ietf-nat-protocol-complications-00.tx. *Work in Progress, Internet Engineering Task Force*, 1999 [0046]
- **J. Lo ; M. Borella.** Real Specific IP: A Framework, draA-ietf-nat-rsip-framework-00.txt. *Work in Progress, Internet Engineering Task Force*, 1999 [0046]
- **Y. Rekhter.** Implications of NATs on the TCP/IP architecture, draft-ietf-nat-arch-implications-00.txt. *Internet Engineering Task Force*, 1999 [0046]
- **J. Postel.** User Datagram Protocol, RFC 768. *Internet Engineering Task Force*, 1980 [0046]
- **J. Postel.** Internet Protocol, RFC 791. *Internet Engineering Task Force*, 1981 [0046]
- **J. Postel.** Transmission Control Protocol, RFC 793. *Internet Engineering Task*, 1981 [0046]
- **J. Postel ; J. Reynolds.** File Transfer Protocol, RFC 959. *Internet Engineering Task Force*, 1985 [0046]
- **R. Braden ; D. Borman ; C. Partridge.** Computing the Internet checksum. *RFC 1071, Internet Engineering Task Force*, 1988 [0046]
- **B. Kantor.** Internet protocol encapsulation of AX.25 frames, RFC 1226. *Internet Engineering Task Force*, 1991 [0046]
- **D. Provan.** Tunneling IPX traffic through IP networks, RFC 1234. *Internet Engineering Task Force*, 1991 [0046]
- **R. Woodburn ; D. Mills.** Scheme for an internet encapsulation protocol: Version 1, RFC 1241. *Internet Engineering Task Force*, 1991 [0046]
- **R. Rivet.** The MD5 message-digest algorithm, RFC 1321. *Internet Engineering Task Force*, 1992 [0046]
- **P. Tsuchiya.** Mutual Encapsulation Considered Dangerous, RFC 1326. *Internet Engineering Task Force*, 1992 [0046]
- **K. Egevang ; P. Francis.** The IP Network Address Translator (NAT), RFC 1631. *Internet Engineering Task Force*, 1994 [0046]
- **S. Hanks ; T. Li ; D. Farinacci ; P. Traina.** Generic Routing Encapsulation, RFC 1701. *Internet Engineering Task Force*, 1994 [0046]
- **S. Hanks ; T. Li ; D. Farinacci ; P. Traina.** Generic Routing Encapsulation over IPv4 networks, RFC 1702. *Internet Engineering Task Force*, 1994 [0046]
- **W. Simpson.** IP in IP Tunneling, RFC 1853. *Internet Engineering Task Force*, 1995 [0046]
- **C. Perkins.** IP Encapsulation within IP, RFC 2003. *Internet Engineering Task Force*, 1996 [0046]
- **C. Perkins.** IP Encapsulation within IP, RFC 2004. *Internet Engineering Task Force*, 1996 [0046]
- **K. Hamzeh.** Ascend Tunnel Management Protocol, RFC 2107. *Internet Engineering Task Force*, 1997 [0046]
- **G. Montenegro.** Reverse Tunneling for Mobile IP, FC 2344. *Internet Engineering Task Force*, 1998 [0046]
- **P. Srisuresh ; D. Gan.** Load Shearing using IP Network Address Translation (LSNAT), RFC 2391. *Internet Engineering Task Force*, 1998 [0046]
- **S. Kent ; R. Atkinson.** Security Architecture for the Internet Protocol, RFC 2401. *Internet Engineering Task Force*, 1998 [0046]
- **S. Kent ; R. Atkinson.** IP Authentication Header, RFC 2402. *Internet Engineering Task Force*, 1998 [0046]
- **S. Kent ; R. Atkinson.** IP Encapsulating Security Payload, RFC 2406. *Internet Engineering Task Force*, 1998 [0046]
- **D. Piper.** The Internet IP Security Domain of Interpretation for ISAKMP. RFC 2407. *Internet Engineering Task Force*, 1998 [0046]
- **D. Maughan ; M. Schertler ; M. Schneider ; J. Turner.** Internet Security Association and Key Management Protocol (ISAKMP), RFC 2408. *Internet Engineering Task Force*, 1998 [0046]
- **D. Hakens ; D. Carrel.** The Internet Key Exchange (IKE), RFC 2409. *Internet Engineering Task Force*, 1998 [0046]

EP 2 254 311 B1

- **A. Conta ; S. Deering.** Generic Packet Tunneling in IPv6 Specification, RFC 2473. *Internet Engineering Task Force*, 1998 **[0046]**
- **B. Carpenter ; C. Jung.** Transmission of IPv6 over IPv4 Domains without Explicit Tunnels, RFC 2529. *Internet Engineering Task Force*, 1999 **[0046]**
- **P. Srisuresh.** IP Network Address Translator (NAT) Terminology and Considerations, draft-ietf-nat-terminology-01.txt. *Work in Progress, Internet Engineering Task Force*, 1998 **[0046]**
- **P. Srisuresh.** Security Model for Network Address Translator (NAT) Domains, draft-ietf-nat-security-01.txt. *Work in Progress, Internet Engineering Task Force*, 1998 **[0046]**
- **P. Srisuresh ; K. Egevang.** Traditional IP Network Address Translator (Traditional NAT), draft-ietf-nat-traditional-01.txt. *Work in Progress, Internet Engineering Task Force*, 1998 **[0046]**
- **W. Teo ; S. Yeow ; R Singh.** IP Relocation through twice Network Address Translators (RAT), draft-ietf-nat-mat-00.txt. *Work in Progress, Internet Engineering Task Force*, 1999 **[0046]**